Flow Networks

- A flow network is a directed graph $G = (V, E)$ with the following characteristics
  - There are two special vertices s and t in V, called the source and sink, respectively
  - Each edge (u, v) in E has an associated nonnegative capacity $c(u, v)$
  - We do not allow any self-loops in E
  - It is convenient to assume that if edge (u, v) belongs to E, then so does edge (v, u)
    - Where necessary, we introduce such an edge (v, u) with $c(v, u) = 0$

Flows

- A flow f in a given flow network $G = (V, E)$ assigns a nonnegative value $f(u, v)$ to each edge (u, v) in E such that the following conditions are satisfied
  - Capacity constraints: For each edge (u, v) in E, we have $0 \leq f(u, v) \leq c(u, v)$
  - Flow conservation constraints: For each vertex v in $V \setminus \{s, t\}$ we have
  $$\sum_{(u,v)\in E} f(u, v) = \sum_{(v,u)\in E} f(v, u)$$
- For any subsets X and Y of V, let $f(X, Y)$ denote
$$\sum_{(u,v)\in E:u\in X,v\in Y} f(u, v)$$

Cuts

- A cut in a given flow network $G = (V, E)$ is a partition of V into an ordered pair of sets (S, T) such that s belongs to S and t belongs to T
- How many cuts does G have?

The Net Flow Across a Cut

- Let f be a flow and let (S, T) be a cut in a given flow network $G = (V, E)$
- We define the net flow of f across cut (S, T) as

$$f(S, T) - f(T, S)$$

Comparing The Net Flow Across Two "Similar" Cuts

- Let f be a flow and let (S, T) and (S', T') be two cuts in a given flow network $G = (V, E)$ such that $|S \oplus S'| = 1$
- Lemma 1: The net flow of f across (S, T) is equal to the net flow of f across (S', T')
  - Let v denote the lone vertex in $S \oplus S'$, and assume without loss of generality that $S' = S + v$

$$
\begin{aligned}
& f(S, T) - f(T, S) \\
= \quad & (f(S + v, T) - f(\{v\}, T)) - (f(T, S + v) - f(T, \{v\})) \\
= \quad & f(S', T) - f(T, S') - f(\{v\}, T) + f(T, \{v\}) \\
= \quad & (f(S', T - v) + f(S', \{v\})) - (f(T - v, S') + f(\{v\}, S')) - \\
& \quad f(\{v\}, T') + f(T', \{v\}) \\
= \quad & f(S', T') - f(T', S') - f(\{v\}, V) + f(V, \{v\}) \\
= \quad & f(S', T') - f(T', S')
\end{aligned}
$$

Comparing The Net Flow Across Distinct Cuts

- Let f be a flow and let (S, T) and (S', T') be two cuts in a given flow network $G = (V, E)$
- Lemma 2: The net flow of f across (S, T) is equal to the net flow of f across (S', T')
  - Follows from repeated application of Lemma 1

The Value of a Flow

- Let f be a flow in a given flow network $G = (V, E)$
- We can define the value of flow f as the net flow out of s
  - This is equal to the net flow of f across cut $(\{s\}, V - s)$
  - By Lemma 2, this is equal to the net flow of f across any cut (S, T)

The Maximum Flow Problem

- The maximum flow problem asks us to determine a flow of maximum value in a given flow network G
- Such a flow is said to be a maximum flow in G
- We will eventually prove the existence of a maximum flow in G
  - A priori, it is unclear whether a maximum flow is guaranteed to exist, even in the special case of integer capacities

The Capacity of a Cut

- Let (S, T) be a cut in a given flow network $G = (V, E)$
- The capacity of cut (S, T) is defined as

$$\sum_{(u,v)\in E:u\in S,v\in T} c(u, v)$$

The Minimum Cut Problem

- The minimum cut problem asks us to determine a cut of minimum capacity in a given flow network G
- Such a cut is said to be a minimum cut in G
- Since there are only a finite number of cuts, a minimum cut is guaranteed to exist

"Max Flow $\leq$ Min Cut"

- Let f be a flow and (S, T) be a cut in a given flow network $G = (V, E)$
- The net flow of f across cut (S, T) is at most the capacity of cut (S, T)
  - The net flow of f across cut (S, T) is $f(S, T) - f(T, S)$, which is at most $f(S, T)$
  - The capacity of cut (S, T) is

$$\sum_{(u,v) \in E : u \in S, v \in T} c(u, v) \geq \sum_{(u,v) \in E : u \in S, v \in T} f(u, v) = f(S, T)$$

- Thus the value of f is at most the capacity of any cut
- Thus the value of a maximum flow is at most the capacity of a minimum cut

Ford-Fulkerson Algorithm: High-Level Plan

- We wish to compute a maximum flow (and also a minimum cut) in a given flow network
- Starting with the "all-zeros flow" (which is feasible), we will iteratively apply two techniques
  - A technique to determine whether a given flow network admits a flow of positive value, and if so, to find such a flow
  - Given a flow f for a flow network G, a technique to construct a "residual network" G_f that faithfully models the "leftover capacity" in G after introducing flow f
    - In particular, the "sum" of f and a maximum flow in G_f should correspond to a maximum flow in G

Determining a Flow of Positive Value

- We wish to determine whether a given flow network $G = (V, E)$ admits a flow of positive value, and if so, to find such a flow
- We can use breadth-first search (BFS) or depth-first search (DFS) to determine (in linear time) whether there is a (simple) directed path P of positive-capacity edges from s to t
- If so, we can obtain a flow f of positive value in G
  - Let Δ denote the minimum capacity of any edge on P
  - Set $f(u, v)$ to Δ for each edge on P, and to zero for all other edges
  - The value of flow f is Δ
- What if there is no such path P?

Determining that the Value of a Maximum Flow is Zero

- Let S denote the set of all vertices v such that there is a directed path of positive-capacity edges from s to v in G, and assume that the sink t does not belong to S
- Let T denote $V \setminus S$
- Observe that (S, T) is a cut in G with capacity zero
- Hence the value of a maximum flow in G is zero
- Thus the all-zeros flow is a maximum flow in G

The Residual Network with Respect to a Flow

- Let f be a flow in a given flow network $G = (V, E)$
- We define another flow network $G_f(V, E)$, called the residual network of G with respect to f
  - Flow networks G and G_f differ only in terms of the edge capacities
- For any edge (u, v) in E, we define the residual capacity of (u, v), denoted $c_f(u, v)$, as $c(u, v) - f(u, v) + f(v, u)$
  - Note that $c_f(u, v) \geq 0$ since $f(u, v) \leq c(u, v)$ and $f(v, u) \geq 0$
  - Note that $c_f(u, v) + c_f(v, u) = c(u, v) + c(v, u)$

"Canonical" Flows

- Let $G = (V, E)$ be a given flow network and let 0 denote the all-zeros flow in G
- For any two functions g and h mapping E to the nonnegative reals, we define $g \oplus h$ as the function that maps each edge (u, v) in E to $f(u, v) - \min(f(u, v), f(v, u))$ where $f = g + h$ (pointwise sum)
  - Note that $\oplus$ is associative and commutative, and $f \oplus g = f \oplus g \oplus 0$
- Let f be a flow in G
  - Note that $f \oplus 0$ is a flow in G and $\text{value}(f \oplus 0) = \text{value}(f)$
  - We say that f is canonical if $f = f \oplus 0$
  - Equivalently, f is canonical if for any edge (u, v) in E, we have $f(u, v) = 0$ or $f(v, u) = 0$

The Residual Network Models the Leftover Capacity

- Let f be a flow in a given flow network $G = (V, E)$
- Lemma 3: If f' is a flow in the residual network G_f then $f'' = f \oplus f'$ is a flow in G and $\text{value}(f'') = \text{value}(f) + \text{value}(f')$
- Let $\hat{f}$ map each edge (u, v) in E to $f(v, u)$
- Lemma 4: If f' is a flow in G, then $f'' = \hat{f} \oplus f'$ is a flow in the residual network G_f and $\text{value}(f'') = \text{value}(f') - \text{value}(f)$

Proof of Lemma 3

- Our main task is to verify that f'' satisfies the upper bound constraints on capacity
  - Assume without loss of generality that f is canonical
  - Let (u, v) and (v, u) be a pair of edges in E, and assume without loss of generality that $f(v, u) = 0$
  - We have $f(u, v) + f'(u, v) \leq f(u, v) + c_f(u, v) = c(u, v) + f(v, u) = c(u, v)$; thus $f''(u, v) \leq c(u, v)$
  - We have $f(v, u) + f'(v, u) = f'(v, u) \leq c_f(v, u) = c(v, u) - f(v, u) + f(u, v) = c(v, u) + f(u, v)$
  - Since $f(v, u) + f'(v, u) \leq c(v, u) + f(u, v)$ and $f(u, v) + f'(u, v) \geq f(u, v)$, the definition of $\oplus$ implies that $f''(v, u) \leq c(v, u)$
- It is easy to see that f'' is nonnegative and satisfies flow conservation, and that $\text{value}(f'') = \text{value}(f) + \text{value}(f')$

Proof of Lemma 4

- Our main task is to verify that f'' satisfies the upper bound constraints on capacity
  - Assume without loss of generality that f is canonical
  - Let (u, v) and (v, u) be a pair of edges in E, and assume without loss of generality that $f(u, v) = 0$
  - We have $\hat{f}(u, v) + f'(u, v) \leq f(v, u) + c(u, v) = c_f(u, v) + f(u, v) = c_f(u, v)$; thus $f''(u, v) \leq c_f(u, v)$
  - Since $\hat{f}(v, u) + f'(v, u) = f'(v, u) \leq c(v, u)$ and $\hat{f}(u, v) + f'(u, v) \geq f(v, u)$, the definition of $\oplus$ implies that $f''(v, u) \leq c(v, u) - f(v, u)$
  - Since $c(v, u) - f(v, u) = c_f(v, u) - f(u, v) = c_f(v, u)$, we have $f''(v, u) \leq c_f(v, u)$
- It is easy to see that f'' is nonnegative and satisfies flow conservation, and that value(f'') = value(f') − value(f)

Relating Maximum Flows in G and G_f

- Lemma 5: If f is a flow in a given flow network G and f' is a maximum flow in the residual network G_f, then $f'' = f \oplus f'$ is a maximum flow in G
  - Lemma 3 implies that f'' is a flow in G and $\text{value}(f'') = \text{value}(f) + \text{value}(f')$
  - Let f_0 be an arbitrary flow in G
  - Lemma 4 implies that $f_0' = \hat{f} \oplus f_0$ is a flow in G_f and $\text{value}(f_0') = \text{value}(f_0) - \text{value}(f)$
  - Since f' is a maximum flow in G_f, we have $\text{value}(f') \geq \text{value}(f_0')$
  - Hence $\text{value}(f'') - \text{value}(f) = \text{value}(f') \geq \text{value}(f_0') = \text{value}(f_0) - \text{value}(f)$
  - Thus $\text{value}(f'') \geq \text{value}(f_0)$

Augmenting Paths

- Let f be a flow in a given flow network $G = (V, E)$
- An augmenting path in the residual network G_f is a directed path of positive-capacity edges from s to t

Characterization of the Set of Maximum Flows

- Let f be a flow in a given flow network $G = (V, E)$
- Lemma 6: Flow f is a maximum flow in G if and only if there is no augmenting path in G_f
  - If there is an augmenting path in G_f, then the value of a maximum flow in G_f is positive and hence Lemma 3 implies that f is not a maximum flow in G
  - If there is no augmenting path in G_f, then the all-zeros flow is a maximum flow in G_f, and Lemma 5 implies that f is a maximum flow in G

Using a Maximum Flow to Compute a Minimum Cut

- Let f be a maximum flow in a given flow network $G = (V, E)$
- Run BFS (or DFS) from s in the subgraph of G_f corresponding to the edges with positive residual capacity
- Let S denote the set of vertices reached
- Let T denote $V \setminus S$, which includes t
- For each edge (u, v) such that u belongs to S and v belongs to T, we have $c_f(u, v) = 0$
  - Hence $f(u, v) = c(u, v)$ and $f(v, u) = 0$
- Thus the capacity of cut (S, T) is equal to the net flow of f across (S, T), which in turn is equal to the value of f
- Since the value of a maximum flow is at most the capacity of a minimum cut, we deduce that (S, T) is a minimum cut

Towards the Max-Flow Min-Cut Theorem

- Let G be a given flow network
- The argument given on the previous slide also shows that if a maximum flow in G exists, then the value of a maximum flow in G is equal to the capacity of a minimum cut in G

The Ford-Fulkerson Maximum Flow Algorithm

- Initialize f to the all-zeros flow
- While there is an augmenting path P in G_f
  - Let $\Delta > 0$ denote the minimum residual capacity of any edge on P
  - Let f_0 denote the flow in G_f that assigns Δ units of flow to each edge on P, and no flow to the remaining edges
  - Update f to $f \oplus f_0$

(Partial) Correctness of the Ford-Fulkerson Algorithm

- Lemma 7: If the algorithm terminates, then it does so with f equal to a maximum flow
  - Initially, f is a flow in G (the all-zeros flow)
  - By Lemma 3, the algorithm maintains the invariant that f is a flow in G
  - The claim follows by Lemma 6

Remark: A Recursive Variant of Ford-Fulkerson

- If there is no augmenting path in G, return the all-zeros flow
- Let P denote an augmenting path in G
- Let $\Delta > 0$ denote the minimum capacity of any edge on P
- Let f denote the flow in G that assigns Δ units of flow to each edge on P, and no flow to the remaining edges
- Recursively compute a maximum flow f' in G_f
- Return $f \oplus f'$

The Special Case of Integer Capacities

- Lemma 8: If the edges capacities are integers, then the Ford-Fulkerson algorithm terminates with an integer maximum flow after a finite number of iterations
  - We can prove by induction on the number of iterations that all of the flow values and residual capacities computed during the course of the algorithm are integers
  - Each iteration increases the value of f by at least 1
  - Since the edge capacities are finite, the value of a maximum flow is finite
- Lemma 8 implies that every integer instance of the maximum flow problem admits an integer maximum flow